RoboTransfer: Controllable Geometry-Consistent Video Diffusion for Manipulation Policy Transfer

Supplementary Material

A. More Experiment Results

A.1. Qualitative Comparison

Figure S1 compares robot data generation methods. Cosmos Transfer [1] performs well under fixed camera views but degrades notably for dynamic viewpoints. In contrast, *RoboTransfer* maintains strong multi-view consistency, producing realistic and coherent novel-view synthesis. RoboEngine [14], based on image inpainting, suffers from noise and jitter, lacks temporal consistency, and cannot precisely control scene backgrounds or objects.

A.2. Diverse Scene Synthesis

For fixed-arm tabletop tasks, backgrounds are simple, while mobile manipulation involves complex geometries. Figure S2 shows that *RoboTransfer* can generate richer, more diverse scenes in complex settings, enhancing the variety and realism of synthetic data for robotic learning.

B. RoboTransfer Implementation Details

B.1. Training Details

Training Dataset. To construct our training dataset, we leverage the open-source Cobot Magic platform to collect a large-scale video corpus of dual-arm robot executions(see Sec. C.1 for details). The raw videos are segmented into 10Hz clips of 30 frames each, resulting in approximately 24k clips for training. Additionally, we curate a set of 1.6k 10Hz 30-frame clips from the collected dataset for video synthesis quality evaluation. All videos are annotated with conditions to facilitate both training and assessment.

Training Details. *RoboTransfer* is fine-tuned from the pre-trained Stable Video Diffusion [2] model. During training, videos from each camera view are resized to a resolution of 640×384. We adopt the AdamW[9] optimizer with a learning rate of 3×10^{-5} and a global batch size of 8, training for a total of 70K steps. During inference, we use the EDM scheduler [6] to perform 30 denoising steps and apply classifier-free guidance.

B.2. Modeling and Conditions Injection Details

Object Condition. We resize each object image to 224×224 and pass it through CLIP's image encoder to obtain a single global feature vector per object. These embeddings are then concatenated and fed into the diffusion model. This design allows for finer-grained control, enabling individual manipulation of each object.

Multi-View Consistency Modeling. Previous methods typically introduce a cross-view module to enhance consistency between views. In contrast, *RoboTransfer* simply concatenates multi-view images, integrating inter-view consistency into global spatial consistency, thereby improving multi-view video modeling. Moreover, this design allows direct loading of pre-trained single-view video generation model weights, without requiring significant modifications to the model architecture.

C. RoboTransfer Dataset Construction Details

For training *RoboTransfer*, we collected a dedicated dataset and designed a construction pipeline (Figure 3 in the main paper). Here, we provide additional implementation details.

C.1. Data Collection

Robot Platform. We built a large-scale robotic demonstration dataset using the Agilex Cobot Magic platform, following standardized protocols [7, 10]. Each demonstration includes synchronized RGB-D streams from three Intel RealSense D435i cameras: two hand-eye views and one overhead view (Figure S3).

Dataset Design. To capture diverse manipulation scenarios, we designed twelve distinct tasks (Figure S4) with variations in objects, backgrounds, and interactions. For each task, 100 demonstration segments were collected across 10 unique object configurations, totaling 1,000 samples per task. Backgrounds range from textured tabletops to cluttered surfaces, and objects vary in shape, size, and material. This ensures high diversity and realism for robust visual policy training. To further enhance background diversity, we incorporate the AgiBot-World dataset [3].

C.2. Geometry Conditions Construction

Depth Conditions. For RGB-D data, raw sensor depth provides accurate metric information but is often noisy or incomplete, particularly on low-reflectivity surfaces. Conversely, simulator-rendered depth is perfect but lacks domain realism. To bridge this gap, we adopt Video Depth Anything (VAD) [4] to generate temporally coherent and spatially complete depth maps. However, VAD predicts *relative* depth, lacking the global metric scale essential for robotic manipulation. We therefore treat the raw sensor depth as a **sparse metric anchor** and the VAD output as a **dense structural prior**. We align them using a robust multi-frame least-squares fitting strategy (Figure S5). Crucially, this

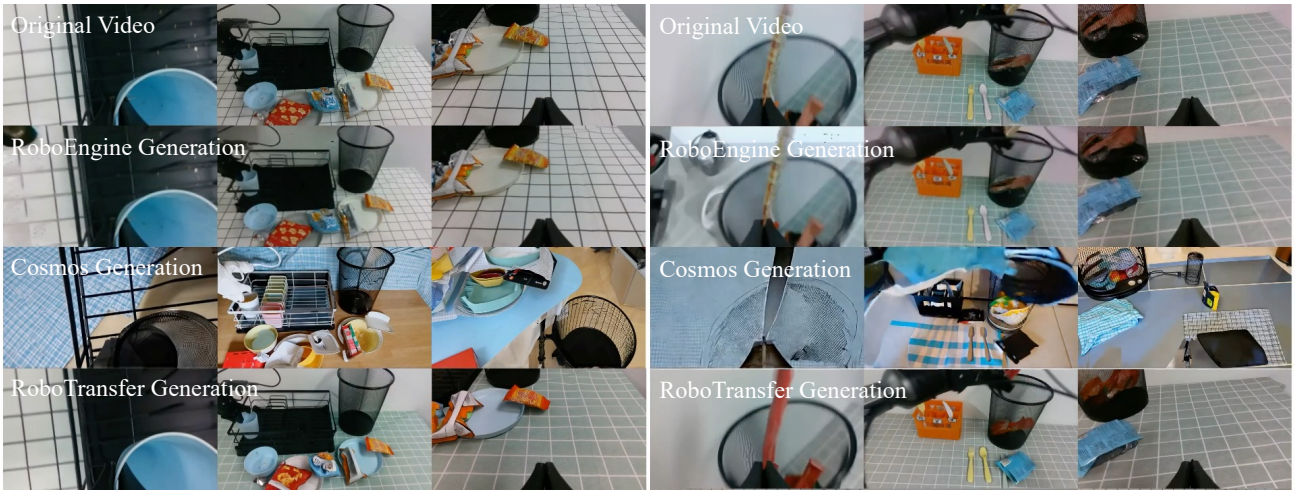

Figure S1. Comparison with Cosmos and RoboEngine generation results.

Figure S2. Visualizations of *RoboTransfer* for diversity scene generation results.

process employs an iterative outlier filtering mechanism (Algorithm 1) that allows the model to learn the correct scale s and shift b from reliable sensor pixels while ignoring sensor noise/holes. This ensures the final output inherits the metric accuracy of the sensor while retaining the completeness of VAD. For datasets without multi-view RGB-D sensors, such as AgiBot-World [3], we estimate metric depth using MoGe [13].

Normal Conditions Surface normals capture fine geometric details and are scale-invariant. We compute per-frame normals using LOTUS [5]. For datasets lacking multi-view RGB-D sensors, MoGe [13] simultaneously estimates depth and normals, streamlining prelabeling.

C.3. Appearance Conditions Construction

Keyframe Selection. Keyframes serve as the reference for object and background appearance. For standard tabletop tasks, we adopt a temporal strategy: the initial frame (fully

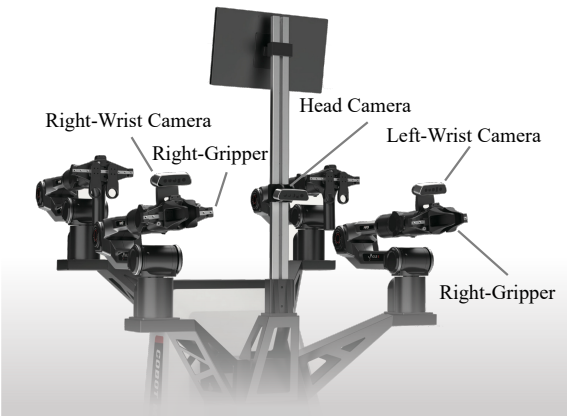

Figure S3. Robot platform visualizations.

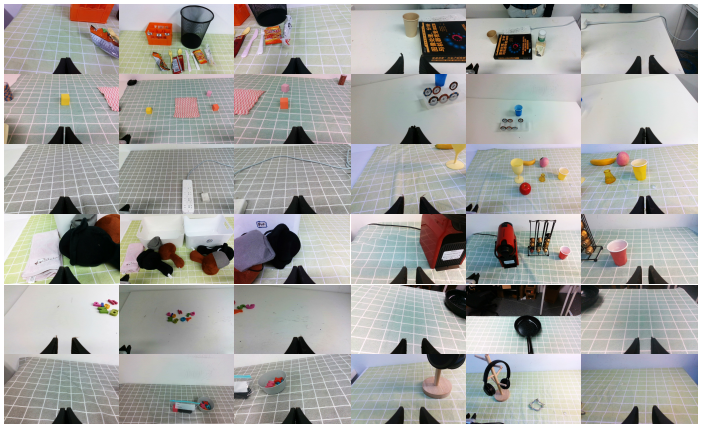

Figure S4. Dataset collected with the Agilex robot.

populated) defines the object condition, while the final frame (post-manipulation) serves as the background. For complex scenarios, we employ an automated content-aware pipeline using VLM descriptors and Grounding DINO. This pipeline selects the frame maximizing the object's visible pixel area for the object condition, and conversely, the frame with minimal object presence for the background condition.

Object Mask Generation. Object descriptions, generated via structured prompts (Figure S6) specifying color, material, shape, and spatial position, are fed into Grounding DINO [8] to generate bounding boxes, which SAM2 [12] converts into object masks.

Objects and Background Conditions. Individual object patches are resized to 224×224 and passed through CLIP [11] to obtain embeddings. Background conditions are generated by masking out all detected objects and applying inpainting to reconstruct the object-free scene.

D. Robot Policy Model Implementation Details

To validate the effectiveness of the data generated by *RoboTransfer*, we train a visual policy using the procedures detailed below for both training and deployment.

D.1. Data Collection and Preprocessing

To ensure a fair evaluation, we excluded the *RoboTransfer* training dataset (Sec. C.1) from the real-robot experiments. Data preparation was conducted as follows:

Real Expert Data We collected 100 expert demonstration sets per manipulation task using the ALOHA teleoperation system. Observations included RGB images at 1280×720 resolution, downscaled to 640×360 for training efficiency, captured at 30Hz, and sampled at 10Hz. Auxiliary robot state information, including joint positions and end-effector poses, was recorded at 200Hz and downsampled to 50Hz for policy training.

Synthetic Data Generation for Policy Fine-tuning. To improve generalization, we generated synthetic videos based on the real demonstrations, introducing variations in foreground objects and background scenes. The pretrained diffusion model conditioned each synthetic video on: (1) per-frame 3D geometry inputs (depth and normal maps) from real demonstrations, and (2) a reference image containing novel objects and backgrounds from a held-out set independent of the policy training data.

D.2. Training Pipeline and Core Parameters

Our training pipeline uses the ACT (Action Chunking Transformer [15]) architecture, processing visual input from three cameras (two wrist-mounted, one overhead). At each timestep, the model receives one RGB frame per view.

Training Objective and Strategy. The policy predicts the next 100 robot states (2s horizon at 50Hz). We adopt a pretrain-then-finetune strategy: 1) **Pretraining on Real Data:** The model was first pretrained for 100k steps using the collected real expert demonstration data. During this phase, the batch size was set to 512, and the learning rate was 1×10^{-4}. 2) **Finetuning with Synthetic Data:** After pretraining, the synthetic data was introduced to fine-tune the model for an additional 50k steps. The learning rate for this phase was reduced to 1×10^{-5}.

All training was performed on a cluster equipped with 8 NVIDIA H20 GPUs. The pretraining phase took approximately 24 hours, and the fine-tuning phase required about 12 hours.

D.3. Real-Robot Deployment and Evaluation

Deployment Platform. Policies were evaluated on the Agilex Cobot Magic platform (Figure S3), the same system used for data collection to ensure consistency between training and evaluation environments.

Inference Procedure During deployment, the policy operates synchronously: the robot executes the full action sequence (100 actions) generated from the previous inference step before capturing new observations. At each decision point, the model receives one RGB frame per camera view along with the current robot state and outputs the next 100

Algorithm 1 Dynamic Mask Alignment

Require: $\mathbf{D}_{\text{pred}}, \mathbf{D}_{\text{sensor}} \in \mathbb{R}^{B\times H\times W}$
Ensure: $\mathbf{D}_{\text{metric}} \in \mathbb{R}^{B\times H\times W}$

Initialize mask:
$\mathcal{M} \leftarrow (\mathbf{D}_{\text{sensor}} > \epsilon) \wedge (\mathbf{D}_{\text{pred}} > \epsilon)$
for $i = 1$ **to** 2 **do**
  SCALE FITTING:
  $s, b \leftarrow \text{ScaleFitting}(\mathbf{D}_{\text{pred}}, \mathbf{D}_{\text{sensor}}, \mathcal{M})$
  $\mathbf{D}_{\text{metric}} \leftarrow s \cdot \mathbf{D}_{\text{pred}} + b$
  MASK UPDATE:
  $\mathcal{E} \leftarrow |\mathbf{D}_{\text{pred}} - \mathbf{D}_{\text{sensor}}| \odot \mathcal{M}$
  $\tau \leftarrow \text{Percentile}_{80}(\mathcal{E}[\mathcal{M} > 0])$
  $\mathcal{M} \leftarrow (\mathcal{E} < \tau) \odot \mathcal{M}$
end for
return $\mathbf{D}_{\text{metric}}$

Algorithm 2 Scale Fitting (Least Squares)

Require: $\mathbf{D}_{\text{pred}}, \mathbf{D}_{\text{sensor}}, \mathcal{M}$
Ensure: $s, b \in \mathbb{R}$

Extract valid pixels:
$\mathbf{p} = \mathbf{D}_{\text{pred}}[\mathcal{M}], \quad \mathbf{s} = \mathbf{D}_{\text{sensor}}[\mathcal{M}]$
Objective:
$\min_{s,b} \|s\mathbf{p} + b\mathbf{1} - \mathbf{s}\|^2$
Closed-form Solution:
$$\begin{bmatrix} s \\ b \end{bmatrix} = \frac{1}{\Delta} \begin{bmatrix} N(\mathbf{p}^\top \mathbf{s}) - (\mathbf{1}^\top \mathbf{p})(\mathbf{1}^\top \mathbf{s}) \\ (\mathbf{p}^\top \mathbf{p})(\mathbf{1}^\top \mathbf{s}) - (\mathbf{p}^\top \mathbf{s})(\mathbf{1}^\top \mathbf{p}) \end{bmatrix}$$
where $\Delta = N(\mathbf{p}^\top \mathbf{p}) - (\mathbf{1}^\top \mathbf{p})^2$
return s, b

Figure S5. **Depth Scale Alignment Algorithms.** We employ an iterative robust least-squares fitting strategy. **Algorithm 1** aligns relative depth predictions to sparse metric sensor data by iteratively filtering high-error regions (outliers). **Algorithm 2** provides the analytical closed-form solution for the linear alignment parameters s (scale) and b (shift).

actions. Inference latency is 10ms, and actions are executed at 50Hz, matching the training robot state frequency.

System Prompt for Object Descriptor:

You are an industrial robotic vision system with 100% detection guarantee. Perform comprehensive scene analysis to identify ALL movable objects, including occluded items. Strictly enforce size constraints and occlusion handling.

Your goal is to complete a list of all visible objects on the table and process the description.

The output Template Format:
A [color] [object], [shape], located [region](x,y)

Examples:
- A red ball, spherical, at the center (250,300);
- A brown chair, angular, in the top-left (100,50);
- A silver-blue arm, mechanical, on the right (600,200);

Figure S6. Visual Description Prompt Template Architecture.

References

[1] Hassan Abu Alhaija, Jose Alvarez, Maciej Bala, Tiffany Cai, Tianshi Cao, Liz Cha, Joshua Chen, Mike Chen, Francesco Ferroni, Sanja Fidler, et al. Cosmos-transfer1: Conditional world generation with adaptive multimodal control. *arXiv preprint arXiv:2503.14492*, 2025. 1

[2] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 1

[3] Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Xindong He, Xu Huang, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025. 1, 2

[4] Sili Chen, Hengkai Guo, Shengnan Zhu, Feihu Zhang, Zilong Huang, Jiashi Feng, and Bingyi Kang. Video depth anything: Consistent depth estimation for super-long videos. *arXiv preprint arXiv:2501.12375*, 2025. 1

[5] Jing He, Haodong Li, Wei Yin, Yixun Liang, Leheng Li, Kaiqiang Zhou, Hongbo Liu, Bingbing Liu, and Ying-Cong Chen. Lotus: Diffusion-based visual foundation model for high-quality dense prediction. *arXiv preprint arXiv:2409.18124*, 2024. 2

[6] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022. 1

[7] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024. 1

[8] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded

pre-training for open-set object detection. In *European Conference on Computer Vision*, pages 38–55. Springer, 2024. 3

[9] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1

[10] Yao Mu, Tianxing Chen, Zanxin Chen, Shijia Peng, Zhiqian Lan, Zeyu Gao, Zhixuan Liang, Qiaojun Yu, Yude Zou, Mingkun Xu, et al. Robotwin: Dual-arm robot benchmark with generative digital twins. *arXiv preprint arXiv:2504.13059*, 2025. 1

[11] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 3

[12] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 3

[13] Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. Moge-2: Accurate monocular geometry with metric scale and sharp details. *arXiv preprint arXiv:2507.02546*, 2025. 2

[14] Chengbo Yuan, Suraj Joshi, Shaoting Zhu, Hang Su, Hang Zhao, and Yang Gao. Roboengine: Plug-and-play robot data augmentation with semantic robot segmentation and background generation, 2025. 1

[15] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023. 3